

1 **CLAIMS**

2
3 1. A processor-readable medium comprising processor-executable
4 instructions configured for:

5 constructing a filter graph to process a data stream from a source file;
6 processing the data stream through the filter graph;
7 receiving an instruction to load a new filter into the filter graph;
8 recognizing the new filter based on registration parameters stored in a
9 registry; and
10 dynamically loading the new filter into the filter graph during the
11 processing.

12
13 2. A processor-readable medium as recited in claim 1, wherein the
14 constructing a filter graph comprises:

15 reading a registry of filter characteristics;
16 identifying a plurality of filters available for operation in a filter graph
17 based on the filter characteristics;

18 creating from the plurality of filters, an instance of a class of filters
19 appropriate for rendering the data stream, each filter in the class of filters operative
20 to conduct a processing operation and having at least one input pin and at least one
21 output pin; and

22 connecting the pins of the filters in the class of filters to assemble the filter
23 graph, the filter graph comprising connected filters wherein the first filter in the
24 filter graph accepts the data stream and the final filter in the filter graph renders the
25 data stream.

1
2 3. A processor-readable medium as recited in claim 1, wherein the
3 dynamically loading comprises:

4 automatically stopping the processing at a current location in the data
5 stream;

6 automatically loading the new filter into the filter graph; and

7 automatically restarting the processing at the current location in the data
8 stream, the processing after the restarting including processing the data stream
9 through the new filter.

10
11 4. A processor-readable medium as recited in claim 3, wherein the
12 automatically loading the new filter comprises:

13 deconstructing the filter graph at a connection point between two filters;

14 inserting the new filter at the connection point; and

15 reconstructing the filter graph such that it includes the new filter inserted
16 between the two filters.

17
18 5. A processor-readable medium comprising processor-executable
19 instructions configured for:

20 receiving a call to register a plug-in; and

21 in accordance with the call, receiving a set of registration parameters
22 comprising:

23 a pwszFriendlyName parameter designating a name for the plug-in;

24 a pwszDescription parameter designating a description of the plug-in;
25

1 a pwszUninstallString parameter designating an uninstall string for
2 uninstalling the plug-in;

3 a dwPriority parameter designating an integer value containing a priority
4 position of the plug-in in a chain of currently enabled plug-ins;

5 a guidPluginType parameter designating a globally unique identifier that
6 specifies a type for the plug-in;

7 a Clsid parameter designating a class identifier of the plug-in;

8 a cMediaType parameter designating a count of media types supported by
9 the plug-in; and

10 a pMediaType parameter designating a pointer to an array of media types
11 that enumerates supported media types for the plug-in.

12
13 6. A processor-readable medium as recited in claim 5, comprising
14 further processor-executable instructions configured for storing the set of
15 registration parameters according to a specific format in a registry of an operating
16 system on a machine wide basis.

17
18 7. A processor-readable medium as recited in claim 6, wherein the
19 specific format comprises:

20 a plug-in type specified by a guidPluginType parameter;

21 a plug-in major format specified by a pMediaType parameter that further
22 specifies the plug-in type;

23 a plug-in minor format specified by another pMediaType parameter that
24 further specifies the plug-in major format;

1 a unique identification of a plug-in specified by a Clsid parameter, the
2 unique identification identifying a plug-in capable of processing the plug-in minor
3 format;

4 a configuration heading for the plug-in type that includes the unique
5 identification of the plug-in specified by the Clsid parameter;

6 a description of the plug-in type specified by the pwszDescription
7 parameter;

8 a name of the plug-in type specified by the pwszFriendlyName parameter;

9 a priority of the plug-in type specified by the dwPriority parameter; and

10 an uninstall path specified by the pwszUninstallString parameter.

11
12 **8.** A processor-readable medium as recited in claim 5, comprising
13 further processor-executable instructions configured for storing a subset of the set
14 of registration parameters according to a specific format in a registry of an
15 operating-system on a per user basis.

16
17 **9.** A processor-readable medium as recited in claim 8, wherein the
18 specific format comprises:

19 a configuration heading for the plug-in type that includes the unique
20 identification of the plug-in specified by the Clsid parameter;

21 an enable indicator permitting the user to enable the plug-in; and

22 a priority of the plug-in type specified by the dwPriority parameter.

1 **10.** A processor-readable medium as recited in claim 5, comprising
2 further processor-executable instructions configured for:
3 constructing a filter graph to process a data stream from a source file;
4 processing the data stream through the filter graph;
5 receiving an instruction to load the plug-in;
6 searching the registry for the plug-in;
7 recognizing the plug-in based on the set of registration parameters stored in
8 the registry; and
9 dynamically loading the plug-in into the filter graph during the processing.

10
11 **11.** A processor-readable medium as recited in claim 10, wherein the
12 dynamically loading comprises:
13 automatically stopping the processing at a current location in the data
14 stream;
15 automatically loading the plug-in into the filter graph; and
16 automatically restarting the processing at the current location in the data
17 stream, the processing after the restarting including processing the data stream
18 through the plug-in.

19
20 **12.** A processor-readable medium as recited in claim 11, wherein the
21 automatically loading the plug-in comprises:
22 deconstructing the filter graph at a connection point between two filters;
23 inserting the plug-in at the connection point; and
24 reconstructing the filter graph such that it includes the plug-in inserted
25 between the two filters.

1
2 13. A processor-readable medium as recited in claim 10, wherein the
3 constructing a filter graph comprises:

4 reading a table of filter characteristics;

5 identifying a plurality of filters available for operation in a filter graph
6 based on the reading;

7 creating from the plurality of filters, an instance of a class of filters
8 appropriate for rendering the data stream, each filter in the class of filters operative
9 to conduct a processing operation and having at least one input pin and at least one
10 output pin; and

11 connecting the pins of the filters in the class of filters to assemble the filter
12 graph, the filter graph comprising connected filters wherein the first filter in the
13 filter graph accepts the data stream and the final filter in the filter graph renders the
14 data stream.

15
16 14. A processor-readable medium as recited in claim 5 having stored
17 thereon a data structure providing a format for storing the registration parameters,
18 the data structure comprising:

19 a PLUG-IN_TYPE field configured to contain a registration parameter;

20 a PLUG-IN_MAJOR_FORMAT field configured to contain a registration
21 parameter;

22 a PLUG-IN_MINOR_FORMAT field configured to contain a registration
23 parameter;

24 a PLUG-IN_TYPE_CONFIGS field configured to contain a registration
25 parameter;

1 a PLUG-IN_ID field configured to contain a registration parameter;
2 a DESCRIPTION field configured to contain a registration parameter;
3 a NAME field configured to contain a registration parameter;
4 a PRIORITY field configured to contain a registration parameter; and
5 a UNINSTALLPATH field configured to contain a registration parameter.
6

7 **15.** A processor-readable medium as recited in claim 5 having stored
8 thereon a data structure providing a format for storing the registration parameters,
9 the data structure comprising:

10 a PLUG-IN_TYPE_CONFIGS field configured to contain a registration
11 parameter;
12 a PLUG-IN_ID field configured to contain a registration parameter;
13 an ENABLED field configured to contain a registration parameter; and
14 a PRIORITY field configured to contain a registration parameter.
15

16 **16.** A processor-readable medium having stored thereon a data structure
17 that describes registration information for a media player to recognize a plug-in,
18 the data structure comprising:

19 a pwszFriendlyName parameter designating a plug-in name;
20 a pwszDescription parameter designating a plug-in description;
21 a pwszUninstallString parameter designating an uninstall string for
22 uninstalling a plug-in;
23 a dwPriority parameter designating an integer value containing a priority
24 position of the plug-in in a chain of currently enabled plug-ins;
25

1 a guidPluginType parameter designating a globally unique identifier that
2 specifies a type for the plug-in;

3 a Clsid parameter designating a class identifier of the plug-in;

4 a cMediaTypes parameter designating a count of media types supported by
5 the plug-in; and

6 a pMediaTypes parameter designating a pointer to an array of media types
7 that enumerates supported media types.

8
9 17. A processor-readable medium as recited in claim 16, wherein the
10 data structure is stored in an area of the processor-readable medium that is an
11 operating system registry.

12
13 18. A computer comprising the processor-readable medium as recited in
14 claim 16.

15
16 19. A processor-readable medium having stored thereon a data structure,
17 comprising:

18 a PLUG-IN_TYPE field containing data indicating a plug-in type;

19 a PLUG-IN_MAJOR_FORMAT field containing data indicating a subset of
20 the plug-in type;

21 a PLUG-IN_MINOR_FORMAT field containing data indicating a second
22 subset of the plug-in type;

23 a PLUG-IN_TYPE_CONFIGS field containing data indicating a
24 configuration of the plug-in type;

1 a PLUG-IN_ID field containing data indicating a globally unique vendor
2 identification of the plug-in type;
3 a DESCRIPTION field containing data indicating a description of the plug-
4 in type;
5 a NAME field containing data indicating a name of the plug-in type;
6 a PRIORITY field containing data indicating a priority for the plug-in type;
7 and
8 an UNINSTALLPATH field containing data indicating a string to uninstall
9 the plug-in type.
10

11 **20.** A processor-readable medium having stored thereon a data structure,
12 comprising:

13 a PLUG-IN_TYPE_CONFIGS field containing data indicating a
14 configuration of a plug-in type;
15 a PLUG-IN_ID field containing data indicating a globally unique vendor
16 identification of the plug-in type;
17 an ENABLED field containing data indicating whether the plug-in type is
18 enabled; and
19 a PRIORITY field containing data indicating a priority for the plug-in type
20 in a playback chain.
21

22 **21.** A method comprising:

23 receiving streaming data;
24 constructing a filter graph based on a data type of the streaming data;
25 processing the streaming data through the filter graph;

1 receiving an instruction to load a new filter into the filter graph;
2 recognizing the new filter based on registration parameters stored in a
3 registry; and
4 dynamically loading the new filter into the filter graph during the
5 processing.

6
7 **22.** A method as recited in claim 21, wherein the constructing a filter
8 graph comprises:

9 reading a registry of filter characteristics;
10 identifying filters available to process the streaming data based on the filter
11 characteristics;

12 creating from the filters, an instance of a class of filters appropriate for
13 rendering the streaming data, each filter in the class of filters operative to conduct
14 a processing operation and having at least one input pin and at least one output pin;
15 and

16 connecting the pins of the filters in the class of filters to assemble the filter
17 graph, the filter graph comprising connected filters wherein the first filter in the
18 filter graph accepts the streaming data and the final filter in the filter graph renders
19 the streaming data.

20
21 **23.** A method as recited in claim 21, wherein the dynamically loading
22 comprises:

23 stopping the processing at a current location in the streaming data;
24 deconstructing the filter graph at a connection point between two filters;
25 inserting the new filter at the connection point;

1 reconstructing the filter graph such that it includes the new filter inserted
2 between the two filters; and

3 restarting the processing at the current location in the data stream, the
4 processing after the restarting including processing the data stream through the
5 new filter.

6
7 24. A method as recited in claim 23, wherein the stopping, the
8 deconstructing, the inserting, the reconstructing and the restarting are performed
9 automatically by a media player.

10
11 25. A computer comprising:
12 a registration function configured to receive a set of registration parameters
13 from a filter plug-in, the registration parameters comprising:

14 a pwszFriendlyName parameter designating a name for the plug-in;
15 a pwszDescription parameter designating a description of the plug-in;
16 a pwszUninstallString parameter designating an uninstall string for
17 uninstalling the plug-in;

18 a dwPriority parameter designating an integer value containing a priority
19 position of the plug-in in a chain of currently enabled plug-ins;

20 a guidPluginType parameter designating a globally unique identifier that
21 specifies a type for the plug-in;

22 a Clsid parameter designating a class identifier of the plug-in;

23 a cMediaType parameter designating a count of media types supported by
24 the plug-in; and

1 a pMediaType parameter designating a pointer to an array of media types
2 that enumerates supported media types for the plug-in.
3

4 26. A computer as recited in claim 25, further comprising a registry that
5 includes the set of registration parameters configured on a machine wide basis
6 according to the following format:

7 a plug-in type specified by a guidPluginType parameter;

8 a plug-in major format specified by a pMediaType parameter that further
9 specifies the plug-in type;

10 a plug-in minor format specified by another pMediaType parameter that
11 further specifies the plug-in major format;

12 a unique identification of a plug-in specified by a Clsid parameter, the
13 unique identification identifying a plug-in capable of processing the plug-in minor
14 format;

15 a configuration heading for the plug-in type that includes the unique
16 identification of the plug-in specified by the Clsid parameter;

17 a description of the plug-in type specified by the pwszDescription
18 parameter;

19 a name of the plug-in type specified by the pwszFriendlyName parameter;

20 a priority of the plug-in type specified by the dwPriority parameter; and

21 an uninstall path specified by the pwszUninstallString parameter.
22

23 27. A computer as recited in claim 25, further comprising a registry that
24 includes a subset of the set of registration parameters configured on a per user
25 basis according to the following format:

1 a configuration heading for the plug-in type that includes the unique
2 identification of the plug-in specified by the Clsid parameter;

3 an enable indicator permitting the user to enable the plug-in; and

4 a priority of the plug-in type specified by the dwPriority parameter;

5
6 **28.** A computer as recited in claim 25, further comprising a media player
7 configured to register a filter plug-in according to the registration parameters.

8
9 **29.** A computer as recited in claim 28, further comprising a filter graph
10 manager configured to construct a filter graph based the registration parameters
11 and on a data type of a data stream received by the media player.

12
13 **30.** A computer as recited in claim 29, further comprising a dynamic
14 plug-in loader configured to automatically stop the filter graph from processing
15 the data stream, determine an enabled filter plug-in based on the registration
16 parameters, deconstruct the filter graph, insert the enabled filter plug-in into the
17 filter graph, and restart the processing of the data stream.

18
19 **31.** A computer comprising:
20 means for constructing a filter graph to process a data stream from a source
21 file;
22 means for processing the data stream through the filter graph;
23 means for receiving an instruction to load a new filter into the filter graph;
24 means for recognizing the new filter based on registration parameters stored
25 in a registry; and

1 means for dynamically loading the new filter into the filter graph during the
2 processing.

3
4 **32.** A computer as recited in claim 31, wherein the means for
5 constructing a filter graph comprises:

6 means for reading a registry of filter characteristics;

7 means for identifying a plurality of filters available for operation in a filter
8 graph based on the filter characteristics;

9 means for creating from the plurality of filters, an instance of a class of
10 filters appropriate for rendering the data stream, each filter in the class of filters
11 operative to conduct a processing operation and having at least one input pin and at
12 least one output pin; and

13 means for connecting the pins of the filters in the class of filters to assemble
14 the filter graph, the filter graph comprising connected filters wherein the first filter
15 in the filter graph accepts the data stream and the final filter in the filter graph
16 renders the data stream.

17
18 **33.** A computer as recited in claim 31, wherein the means for
19 dynamically loading comprises:

20 means for automatically stopping the processing at a current location in the
21 data stream;

22 means for automatically loading the new filter into the filter graph; and

23 means for automatically restarting the processing at the current location in
24 the data stream, the processing after the restarting including processing the data
25 stream through the new filter.

1
2 **34.** A computer as recited in claim 33, wherein the means for
3 automatically loading the new filter comprises:

4 means for deconstructing the filter graph at a connection point between two
5 filters;

6 means for inserting the new filter at the connection point; and

7 means for reconstructing the filter graph such that it includes the new filter
8 inserted between the two filters.

9
10 **35.** A filter graph comprising:

11 a first filter configured to process data by a first process, the first filter
12 processing a first part of a data stream and a second part of the data stream;

13 a second filter loaded into the filter graph after the first part of the data
14 stream is processed by the first filter, the second filter configured to process data
15 by a second process, the second filter processing the second part of the data
16 stream; and

17 a rendering filter configured to render the data stream, the rendering filter
18 rendering the first part of the data stream processed by the first filter and rendering
19 the second part of the data stream processed by both the first filter and the second
20 filter.

21
22 **36.** A media player configured to generate the filter graph as recited in
23 claim 35.

24
25 **37.** A computer comprising the media player recited in claim 36.